ARMY RESEARCH LABORATORY

Design and Functionality of the Graphical Interactive Narrative (Gin) System Version 0.2

by Brent J. Lance, Jonroy Canady, and Kelvin S. Oie

ARL-TR-6076 **August 2012**

Approved for public release; distribution is unlimited.

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5425

ARL-TR-6076 **August 2012**

Design and Functionality of the Graphical Interactive Narrative (Gin) System Version 0.2

Brent J. Lance and Jonroy Canady
Human Research and Engineering Directorate, ARL

Kelvin S. Oie
DCS Corporation

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
August 2012	Final	10/1/10–9/30/11

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Design and Functionality of the Graphical Interactive Narrative (Gin) System Version. 0.2	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
Brent J. Lance, Jonroy Canady, and Kelvin S. Oie*	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
U.S. Army Research Laboratory ATTN: RDRL-HRS-C Aberdeen Proving Ground, MD 21005-5425	ARL-TR-6076
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

*DCS Corporation, Alexandria, VA

14. ABSTRACT

This report describes the design and functionality of version 0.2 of the Graph-theoretic Interactive Narrative (Gin) System, a system intended to increase interactivity of virtual reality environments (VEs) used for human experimentation, while still allowing for necessary experimental constraints to be enforced so that the data can be analyzed.

An initial step towards increasing interactivity is to allow human subjects to control their own navigation through the VE. In order to make this initial step, the Gin system uses the graph theory to model the topography of the VE in an abstract fashion; monitor the movement of subjects through the environment; and add, move, activate, deactivate, and/or remove experimental stimuli so that the experimental subjects are exposed to the stimuli in the way that the experiment designer intends for them to be.

In the long term, the goal of the Gin system is to represent both a environmental topography and experimental context graph, allowing for both increased tractability of experimental scenario development, and increased tractability of experimental data analysis.

15. SUBJECT TERMS

interactive narrative, procedural content, graph theory, virtual reality, psychology, neuroscience

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Brent J. Lance
Unclassified	Unclassified	Unclassified	UU	24	19b. TELEPHONE NUMBER (*Include area code*) (410) 278-5943

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures iv

List of Tables iv

1. Introduction 1

1.1 Background 1

1.2 Gin Overview 2

1.3 Architecture 2

2. Methods 3

2.1 Experimental Domain 3

2.2 Graph Representation 6

2.3 Experimental Subjects and Agents 7

2.4 Experimental Stimuli 7

2.5 Information Stored in the Gin Graph 8

2.6 Message Passing 8

2.7 Coordinates 10

3. Results and Discussion 11

3.1 Example 11

3.2 Implementation 11

4. Conclusions 13

4.1 Evaluation 13

4.2 Interface Improvements 14

4.3 Future Work 14

List of Symbols, Abbreviations, and Acronyms 15

Distribution List 16

List of Figures

Figure 1. Overview for interfacing the Gin system with the desired VE, showing the simulation engine, the Gin module, and the interface used to connect the two. 3

Figure 2. VC crew station, consisting of a 180°field-of-view banner across the top, a 60° field-of-view window on the left hand side, and an overhead map on the right hand side. 4

Figure 3. The virtual metropolitan environment, showing the locations of checkpoints and stimuli, and a hypothetical subject path. 5

Figure 4. A virtual metropolitan environment from an experiment showing the generation of a topographical graph where graph edges represent roads in the environment, and the nodes represent the intersections and endpoints of these roads. 6

Figure 5. Fine-grained detail showing how the message passing system transfers information from the simulation engine into the Gin system. 10

Figure 6. Fine-grained detail showing how the message passing system transfers information from the Gin system back out to the simulation engine 10

List of Tables

Table 1. Current messages templates defined by the Gin message passing system. 9

Table 2. Memory usage of the Gin system as a function of graph size. 12

Table 3. Processing time required for the Gin system using a 100-node graph. 13

1. Introduction

1.1 Background

This report describes the design and functionality of version 0.2 of the Graph-theoretic Interactive Narrative (Gin) System. The purpose of the Gin system is to increase the interactivity and sense of agency for human subjects in virtual environments (VEs) used for human-subject neuroscience, while still allowing for necessary experimental constraints to be enforced so that the data can be analyzed. The result of this is that many VEs used for experimentation are rigidly designed, progressing through the experimental scenario in a highly linear fashion. This strongly limits the subject's agency, their ability to make their own decisions in determining their behavior in the VE, in turn, making it difficult, if not impossible, to study real-world decision making in these VEs.

The goals of the Gin system design and development efforts are to address the two underlying problems driving the lack of subject agency. The first problem is the tractability of scenario development. Allowing the participant to control their decisions and take a broad variety of actions in the environment leads to many complications that make it difficult to enable the experimenter to enforce decisions made about the number, type, and order of stimuli that the subject will experience during the experiment. First, the participant could make decisions such that they miss stimuli important to the success of the experiment. Second, the participant could take actions that cause them to be exposed to the same stimuli repeatedly. Finally, development of the VE, including map generation and the placement of stimuli, is very time consuming. This makes it prohibitively expensive to place individual stimuli such that the participant will be exposed to important stimuli and events without repeating them no matter what actions they take.

The second problem is the tractability of analysis. By limiting the subject's interaction, analyzing the resulting data becomes considerably more conceptually and mathematically tractable. However, conducting experiments within these limitations makes it extremely difficult, if not impossible, to use these environments to study real-world decision-making and behavior. Minimizing the ability of the subject to interact with the environment and to make decisions that affect how the scenario progresses strips any agency and decision-making ability away from the experimental subject.

An initial step towards addressing these problems is to improve the tractability of scenario development while providing the user with an increased sense of agency by allowing them to control their own navigation through the VE. In order to make this initial step, the Gin system uses the concepts of procedural game development (i.e., algorithmically generated video game content) and interactive narrative to model the topography of the environment and ensure that the subject is exposed to needed experimental stimuli.

While the current version of the Gin system does not address the tractability of analysis problem, we intend to use future versions of the Gin system to address this issue through storing contextual information about the state of the VE, the subject, and when an experimental stimulus is displayed. In this way, we hope to be able to begin to quantify and utilize the increased complexity arising from these VEs. In addition, by combining the ability to manipulate scenario events with the storing of contextual information, we would like to be able to use the Gin system to drive the experiment. By doing this, we envision leading the user to experience specific stimuli under specific contexts where additional data is required.

1.2 Gin Overview

The Gin system is an approach to addressing problems of agency and complexity in human-subject experiments performed in VEs. The current approach of the Gin system is to: model the topography of the VE in an abstract fashion; monitor the movement of subjects through the environment; and add, move, activate, deactivate, and/or remove experimental stimuli so that the experimental subjects are exposed to the stimuli in the way that the experiment designer intends for them to be.

In order to abstractly model the VE, the Gin system represents the environment as a graph. Graph theory is a common formal modeling approach to problem representation in mathematics and computer science (West, 2001).[1] Graphical models consist of a set of "nodes" or "vertexes," which are connected by "edges." Both nodes and edges can have properties, labels, or values assigned to them in order to better represent the problem. By modeling a problem as a graph, we allow use of many existing graph theory algorithms and techniques to solve or simplify the problem, as well as existing software packages that implement these algorithms and techniques. It also allows for the potential expansion of the system into a topographical and contextual graph, allowing for both increased tractability of scenario development, and increased tractability of analysis.

1.3 Architecture

The architecture for the Gin system and how it connects with a VE can be seen in figure 1. The Gin system is intended to be used as a module, integrated into and controlling some aspects of a simulation or gaming engine. This simulation engine is shown in the box on the left of figure 1. The simulation engine underlies the desired VE being used for the experiment. It displays graphics to the subject, records the subject's input, and provides additional simulation (such as physics modeling) for the experimental VE. The simulation engine also tracks the subject's movement and provides that information to the Gin system through the Gin interface, or Ginterface. Finally, it displays the experimental stimuli to the experimental subjects when they are at the appropriate locations.

[1] West, D. B. *Introduction to Graph Theory*, Vol. 2; Prentice Hall: Upper Saddle River, NJ, 2001.

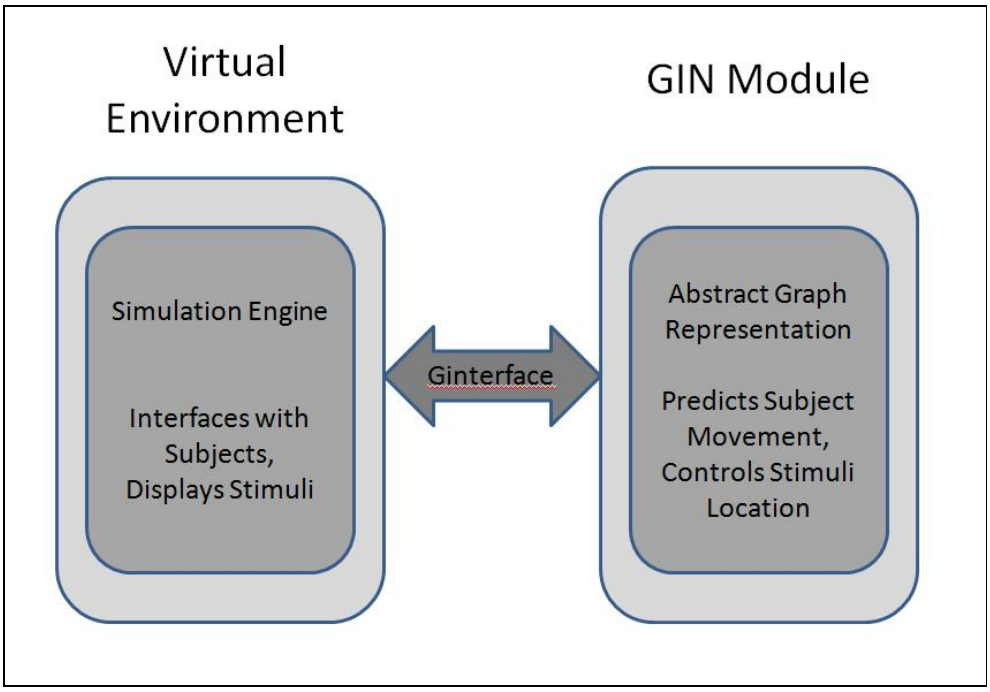

Figure 1. Overview for interfacing the Gin system with the desired VE, showing the simulation engine, the Gin module, and the interface used to connect the two.

In contrast, the Gin system (shown on the right in figure 1) will maintain an abstract model of the topography of the VE, including monitoring the movement of subjects through the environment. It will move, add, delete, activate, and deactivate stimuli based on the locations and expected goals of the subjects in the simulated environment, and it will control the stimuli in the VE by passing messages to the simulation engine through the Ginterface.

2. Methods

2.1 Experimental Domain

The initial experimental paradigm the Gin system is envisioned to be integrated with is a set of joint U.S. Tank Automotive Research Development and Engineering Center (TARDEC)/U.S. Army Research Laboratory (ARL) experiments on decision-making processes while controlling simulated vehicles through Army crew stations in high-complexity VEs (Lance et al., 2011).[2] The Gin system was developed with the express intent of easing the development of the highly complex simulated environments required for these experiments. In 2011, we ran an experiment demonstrating the issues, which the Gin system is intended to address. During the experiment,

[2]Lance, B.; Gordon, S.; Vettel, J.; Johnson, T.; Paul, V.; Manteuffel, C.; Jaswa, M.; Oie, K. Classifying High-noise EEG in Complex Environments for Brain-computer Interaction Technologies. In *Proceedings of the 4th International Conference on Affective Computing and Intelligent interaction-Volume Part II*, 2011, pp 467–476.

teams of two Soldiers performed six simulated missions consisting of traveling in a Stryker vehicle from a Forward Operating Base (FOB) to a nearby small desert metropolitan area, visiting three sequential checkpoints in the city area, and then returning to the FOB. One Soldier was assigned the role of the Vehicle Commander (VC), while the other Soldier was assigned to be the Driver. Each Soldier would spend one day as VC and one day as the Driver. The simulated Stryker was equipped with a closed-hatch Local Situational Awareness (LSA) system, consisting of six external cameras covering a 360° area around the vehicle that were accessible from the VC crew station (figure 2).

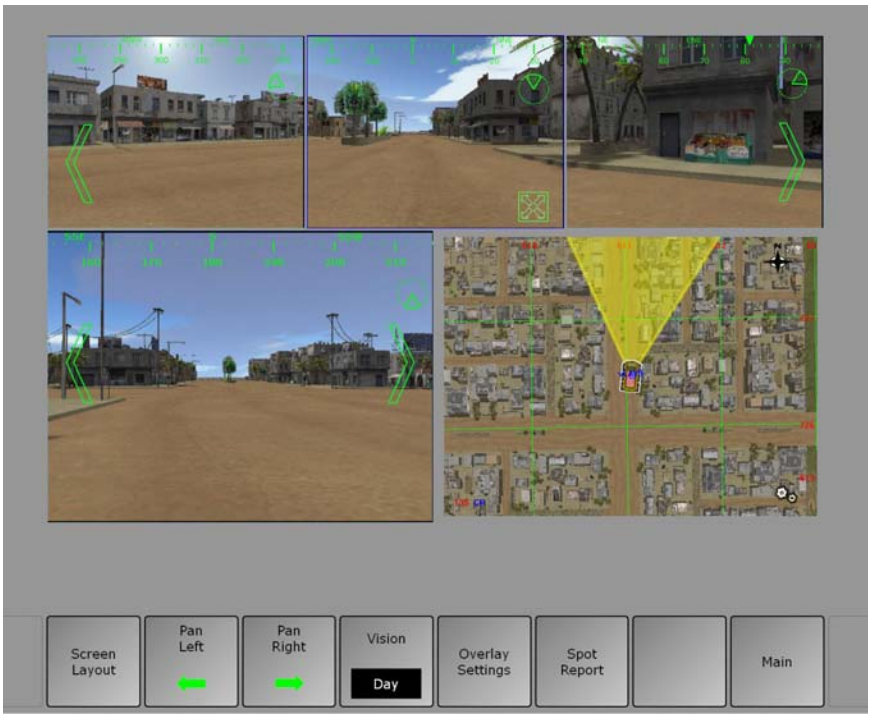

Figure 2. VC crew station, consisting of a 180°field-of-view banner across the top, a 60° field-of-view window on the left hand side, and an overhead map on the right hand side.

During each mission, the VC performed numerous tasks that can be categorized into three main task groupings:

1. Overseeing mission progress and ensuring that the vehicle arrived at each checkpoint within a specific time range. This included supervising the Driver and providing turn-by-turn directions through the city, halt/resume commands, and immediate command driving around difficult obstacles in the environment.

2. Maintaining visual LSA, which included detecting road obstacles and traffic conditions relevant to navigating the environment, and reporting the position of uniformed local forces and objects identified as threats over the radio network to a simulated Tactical Operating Commander (TOC).

3. Maintaining auditory LSA, which included monitoring and responding to radio communications about mission status from the TOC and verbally interacting with the Driver.

For this experiment, the stimuli, specifically, stationary three-dimensional models of Iraqi Army Soldiers standing alongside the road were placed into the environment by hand (see figure 3). As subjects were allowed to take their own paths through the environment, they encountered differing numbers and sets of stimuli. While we attempted to ensure that certain stimuli would be seen by all subjects by placing those stimuli near checkpoints in the environment, different subjects approached the checkpoints from different directions, allowing subjects to unintentionally avoid even these stimuli.

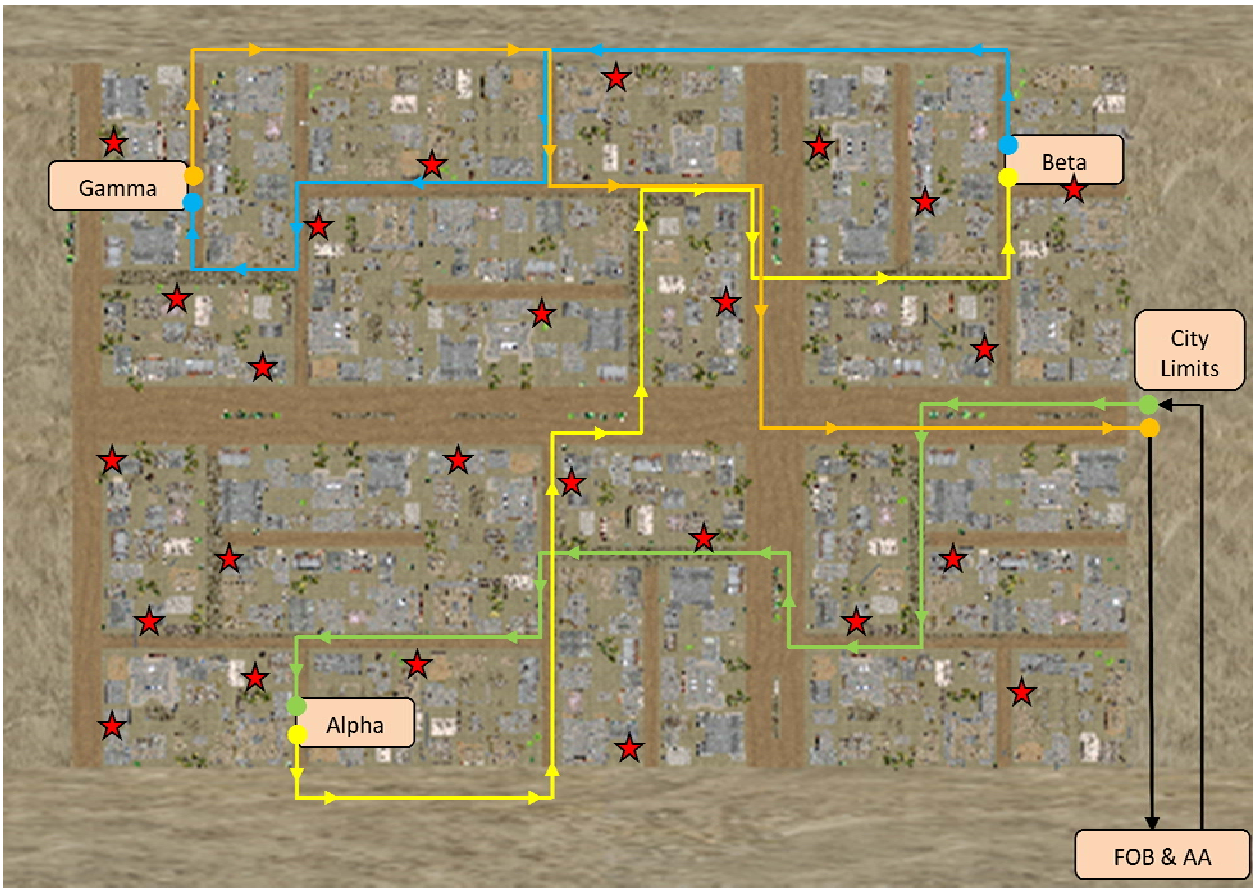

Figure 3. The virtual metropolitan environment, showing the locations of checkpoints and stimuli, and a hypothetical subject path.

2.2 Graph Representation

In order to abstractly represent the VE developed for this experimental domain, the nodes and edges of the graph for the Gin system must represent topographical regions in the simulated environment. In these circumstances, the most-likely graphical model of the environment would have the graph nodes and edges representing the intersections and the roads, respectively, in the VE that the subject's vehicle travels within (see figure 4). However, in order for the Gin system to function, it must represent more aspects of the environment than just the topography. It must also represent the experimental subject, including potential goals that the subject must visit as part of the experiment, as well as the locations and activation status of experimental stimuli in the environment.

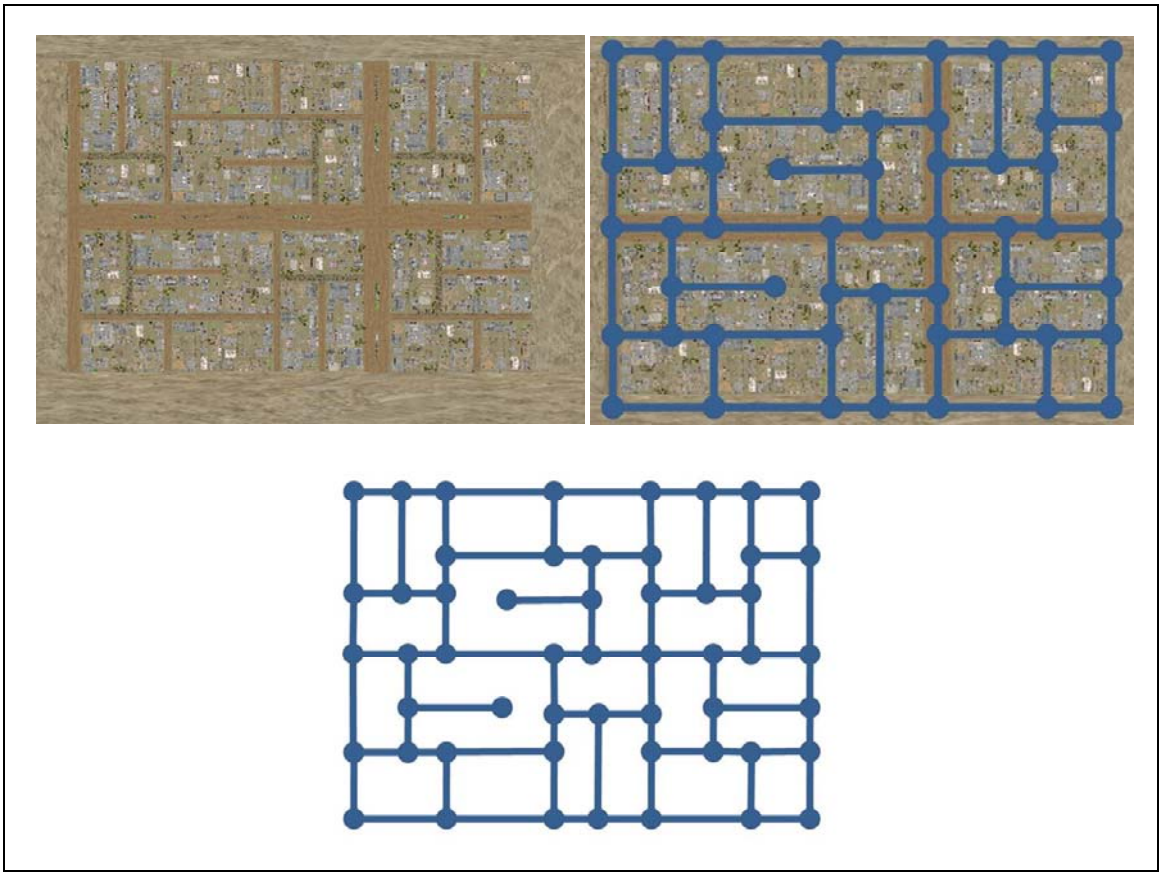

Figure 4. A virtual metropolitan environment from an experiment showing the generation of a topographical graph where graph edges represent roads in the environment, and the nodes represent the intersections and endpoints of these roads.

2.3 Experimental Subjects and Agents

Because the Gin system must monitor the position of the experimental subjects, they must be represented in the graph. Subjects are currently represented as generic "agent" objects, which would also model simulated agents, such as allies or Opposing Forces (OPFOR) in the environment. Essentially, an agent is defined as anything that moves through the environment and is not under the direct control of the Gin system.

Each agent has an ordered list of goals, which are locations in the environment that it is expected to be moving to. These goals are used by the Gin system to predict the movement of the agent through the environment and ensure that stimuli are appropriately presented to the correct agents. Currently, one of the agent's goals is "active" at any given time, and the Gin system assumes that the agent is attempting to move to the location of its active goal. These goals can be predefined as checkpoints in the environment that the subject must visit or placed at opposite ends of the environment.

2.4 Experimental Stimuli

Because the Gin system is controlling the placement of stimuli in the environment, experimental stimuli must be represented in the graph, and they are represented currently as "stimulus" objects. These stimulus objects contain a list of all locations in the graph where they are placed, and a Boolean value for each location indicating if the stimulus instance at that location is active. Active stimuli are displayed to an agent when the agent moves through the location on the graph where the active stimulus is located, while inactive stimuli are not displayed to agents. Stimulus objects are divided into two types, "mobile stimuli," and "immobile stimuli."

Mobile stimuli are placed along the subject's path in real time, based on the current location of the subject and the location of its final goal. To predict the subject's path, the Gin system determines the shortest path from each node neighboring the location of the subject to the location of the agent's final goal. The stimuli are then laid along the edges of the shortest path in a "reverse greedy" fashion, where the last stimuli appears closest to the goal node, and the stimuli are placed along the path in equal intervals until all of the stimuli have been placed. The inter-stimulus interval is determined by dividing the total length of the path by the total number of stimuli (both those that have already been seen and those that have yet to be seen). While it may seem more intuitive to generate the inter-stimulus interval by dividing the path length by only the number of as-yet unseen stimuli, in practice this tends to result in the subject seeing all of stimuli very early in the experiment. This is particularly true if the subject does any backtracking.

Immobile stimuli are placed on the graph during the scenario development process before the user interacts with the crew station. In contrast with mobile stimuli, of which a limited number are placed on the graph and moved as the subject moves through the environment, large numbers of immobile stimuli are placed on all possible paths (identified using a bounded depth-first

search [DFS]) the subject could take through the environment. The algorithm then lays stimuli along each path identified by the DFS, using the same "re/verse greedy" stimulus placement algorithm used to place mobile stimuli. The system then allows the immobile stimuli to be activated or deactivated as needed. This ensures that only one immobile stimulus is active at any given time, depending on what path the subject takes through the environment, and that the user sees all of the stimuli in the correct order without repeats.

The purpose of allowing these two types of stimuli is to enable the user of the system to make a memory/processor tradeoff, if necessary. The mobile stimuli option requires many fewer stimuli to be placed in the environment, but requires considerable computation to be spent on real-time prediction of the subject's path. In contrast, the immobile stimuli option performs the path prediction offline before the experiment occurs, but requires much more memory to be spent placing large numbers of stimuli into the environment.

2.5 Information Stored in the Gin Graph

In order to represent the necessary components of the simulated environment in the Gin graph, we make several changes to the underlying graph, placing the necessary information within the core graph, as well as within the individual nodes and edges. An associative array of agent objects is added to the graph, containing all of the agents that could be moving around the graph at any given time. In addition, an associative array of stimulus objects is added to the graph, containing all of the mobile and immobile stimuli that will be displayed to the subject as they move through the graph. Finally, two lists of stimuli identifiers are added to the graph, indicating the proper orders in which both the mobile and immobile stimuli should be presented to the subject. Currently, only one instance of a specific agent or stimulus object can be in a specific place on the graph at a time.

In addition, a "GinNode" object is added to each node in the graph, and a "GinEdge" object is added to each edge. These objects contain a list of the active and inactive stimuli present in the graph location, a list of the agents currently in the graph location, and the coordinates and extents of the graph location in the simulated environment.

2.6 Message Passing

In order to improve the ability to easily integrate the Gin system and the VE, we have included a custom-built message passing system in the Gin package (see table 1). This message passing system is used to initialize, control, communicate with, and obtain outputs from the Gin system and consists of a set of message templates, an interface that actually passes the messages into and out of the Gin module, a handler function for each message template, and wrapper functions (with "In_" and "Out_" prefixes) that simplify the instantiation of messages from the message templates.

Table 1. Current messages templates defined by the Gin message passing system.

Message	Purpose
AddAgent	Sent to the Gin graph when an agent needs to be added.
UpdateAgentLocation	Sent to the Gin graph when an agent moves in the simulation.
UpdateAgent	Sent to the Gin graph when an agent changes its status in the simulation.
RemoveAgent	Sent to the Gin graph when an agent needs to be removed.
AddNewStimulus	Sent to the Gin graph when a stimulus needs to be added.
AddStimulusInstance	Sent to the simulation to place a stimulus into the simulated environment.
DisplayStimulus	Sent to the Gin graph when a stimulus has been displayed to an agent.
ActivateStimulus	Sent to the simulation to activate a placed stimulus so that it can be displayed.
DeactivateStimulus	Sent to the simulation to deactivate a placed stimulus so that it will no longer be displayed.
RemoveStimulusInstance	Sent to the simulation to remove a stimulus from the simulated environment.
MoveStimulusInstance	Sent to the simulation to move a stimulus within the simulated environment.
ReadGraph	Sent to the Gin graph, telling it to read an existing graph from a file, and replace the current Gin graph with it.
WriteGraph	Sent to the Gin graph, telling it to write the current graph to a file.
InitializeGraph	Sent to the Gin graph, telling it to initialize with a default graph.
SendGraph	Sent to the Gin graph, telling it to send all of the information it has on agents and Stimuli to the simulation engine.
InitializeGinVariables	Allows initialization and modification of miscellaneous variables required by the Gin system.
PlotGraph	Plots the current graph out to a .png image.

The message templates define an extensible set of messages that are recognized by the program. Messages are generated by using the message template to create a message and then populating that message with the data that needs to be sent. Each different message template will have its own required data fields that will need to be filled before the message is ready to be sent. The message passing interface (the Ginterface) actually passes the messages into and out of the Gin module. Currently, this interface is single-threaded and blocking, meaning that any program that sends a message to the Gin system must wait for a response before it can continue. The message handlers receive the message from the messaging interface and extract the message data and pass it into the Gin module (figure 5). Similarly, the message passing system is used to transfer information out of the Gin module (figure 6).

Finally, the wrapper functions labeled “In_”, and “Out_” minimize the amount of code that has to be written in order to generate and pass commonly required messages into and out of the Gin system. “Out_” functions generate and send a message through the messaging interface, while “In_” functions are used to receive messages sent through the interface.

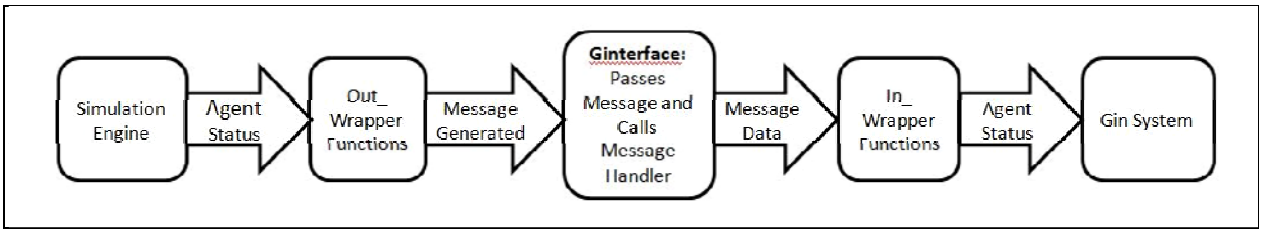

Figure 5. Fine-grained detail showing how the message passing system transfers information from the simulation engine into the Gin system.

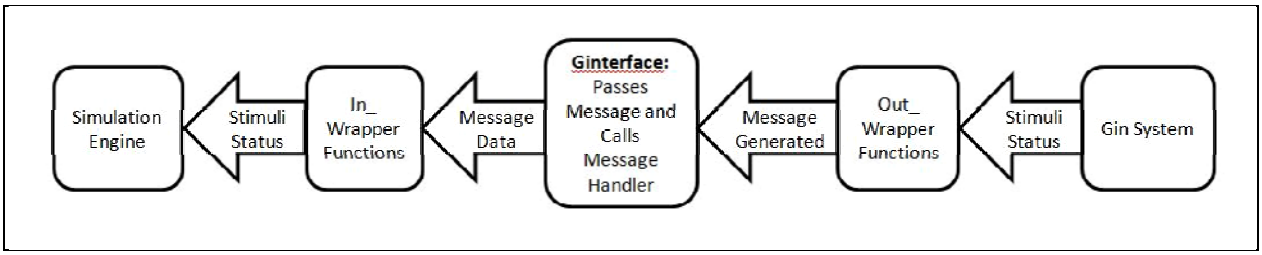

Figure 6. Fine-grained detail showing how the message passing system transfers information from the Gin system back out to the simulation engine.

The message template and handlers provide a communication protocol, which can be easily extended to integrate the Gin system with additional simulation environments. In addition, the message passing interface can be edited and upgraded independently from extending the messages, allowing the interface to be improved while maintaining the same communication protocol.

2.7 Coordinates

As part of this implementation, there are two sets of coordinates for a simulated environment using the Gin system. Both the simulated environment and the abstract graph modeled in the Gin system will have their own coordinates. As a result, the Gin system utilizes a joint Coordinates object that stores coordinates for both the simulation engine and the internal Gin graph in the same object, transforms between them, and provides whichever type of coordinates is needed at any given time. Specifically, given a Coordinates object *c*, the location in the Gin graph can be accessed through *c.gin*, while the location in the simulated environment can be accessed through *c.sim*. The transformation between the two coordinate systems is defined as a function pointer pointing to a default function, allowing the coordinate transformation to be customized for each VE that the Gin system is integrated with.

3. Results and Discussion

3.1 Example

We have implemented a simple example simulated environment into the Gin system for testing purposes and to provide an example of how to integrate the Gin system and a VE. The example simulated environment is a graph that is similar to the internal Gin graph. The example VE simulates an experimental subject as an agent class that is provided with a set of random goals, similar to mission checkpoints in the TARDEC/ARL experiment discussed above. As the simulated subject moves through the example VE, its location is sent to the Gin system, which predicts its movement, places stimuli on the internal Gin graph based on this predicted movement, and sends messages to the example VE, notifying it as to where stimuli should be placed. These messages passed from the example VE describing the agent location, and from the Gin system that describe the locations of stimuli, keep both the simulation engine and the Gin graph synchronized throughout the execution of the simulation (see figures 5 and 6).

3.2 Implementation

Version 0.1 of the system was developed in 2010 (Lance et al., 2010).[3] Gin version 0.1 consisted of an extremely simple set of ad-hoc scripts that, while demonstrating that the initial concept was theoretically sound, would have been impossible to integrate with any simulated environment. In contrast, Gin version 0.2 is a full Python package, allowing the system to be imported and used in many Python programs, and includes documentation, example code, and function tests.

While there are still improvements that need to be made, it could be used for experimentation in its current state, although integration with most simulation engines would be nontrivial. If the simulation engine is compatible with Python, the Gin system package could be simply included into the system. If the simulation engine is not compatible with Python, integrating it will require writing a network wrapper to the message passing system and communicating with the Gin system through a network socket. Planned improvements to the Ginterface will address this integration issue.

The underlying graph implementation used in this package is the NetworkX graph theory library developed by Los Alamos National Labs, and available at http://networkx.lanl.gov. NetworkX is a general-purpose graph-theory and network analysis software package developed by Los Alamos National Laboratories that is designed to allow rapid development of graph-based software projects. In addition to providing a generic way to create, load, and store graph objects,

[3] Lance, B.; Vettel, J.; Paul, V.; Oie, K. The Mission-Based Scenario: Rational and Concepts to Enhance S&T Research Design. In *Modeling & Simulation, Testing and Validation (MSTV) Mini-Symposium,* at NDIA Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), 2010.

NetworkX also provides access to a large number of existing algorithms for analyzing and describing graph objects. Relying on NetworkX for graph creation and manipulation allowed us to develop the Gin system much more rapidly than would have otherwise been possible.

This code is currently stored on a secure network drive at ARL. The code is implemented in Python version 2.6, and based on NetworkX version 1.1. The numerical processing library used for the project is the Python library Numpy, version 1.4. The graphs are visualized using version 0.99 of the Python Matplotlib library, and configuration file processing is performed using PyParsing version 1.5.6.

The unit and function tests for the Gin system consist of 94 test methods, which contain over 475 individual assertion statements that the code must pass. The code currently passes all function tests in the course of ~120–240 s.

Memory usage of the Gin system is shown in table 2. On a 32-bit system with 2 GB RAM, the maximum size of the Gin graph is ~350,000 nodes. However, the size of the currently used VE (see figure 3) is ~53 nodes. Keeping the memory usage of the Gin system under 100 MB allows graphs of over 10,000 nodes, 200× the size needed for the current VE. As a result, it should be possible to represent highly complex VEs while still maintaining a minimal memory footprint.

Table 2. Memory usage of the Gin system as a function of graph size.

Memory Usage (KB)	**Graph Size**
24,080	100 nodes (10 × 10 graph)
76,180	10,000 nodes (100 × 100 graph)
350,012	62,500 nodes (250 × 250 graph)
1,325,384	250,000 nodes (500 × 500 graph)
1,738,012	330,625 nodes (575 × 575 graph
1,882,128	354,025 nodes (595 × 595 graph)
Memory error on 32 bit system	358,801 nodes (599 × 599 graph)

Processing time required for the Gin system is shown in table 3. The test system displayed 20 mobile stimuli and 10 immobile stimuli to a single simulated subject moving to three pre-specified goals on 100 randomly generated 100-node graphs set up as 10 × 10 squares. As shown in table 3, the speed of the Gin system is acceptable for real-time use, taking ~0.2–0.3 s to process each time the agent moved from one location in the graph to another. We would like to note that this performance was achieved with active debugging code on a laptop with a dual-core Intel 2.5 GHz processor and 8 GB of RAM. Changes in either aspect could greatly increase performance.

Table 3. Processing time required for the Gin system using a 100-node graph.

Simulation Time for 100-node graph	Average (s)	Min (s)	Max (s)
Entire simulation	26.530	26.102	28.078
Single step	0.218	0.0000209	0.314

The code for Gin version 0.2 represents a vast improvement over the previous version 0.1 system. Version 0.1 consisted of proof-of-concept ad-hoc scripts, which could not have been used in any system. Version 0.2 is a Python package, with several notable features, including highly general code, allowing for easy expansion to alternative domains; an easily extensible message passing system, allowing the Gin system to be integrated with a variety of environments; and a test-driven development cycle, leading to fewer bugs, easy adoption of major architectural changes, and a system that is easy for a new programmer to learn. While there are still improvements to be made, it is possible to use the Gin system for experimentation in its current state; although integration with most simulation engines would be nontrivial.

4. Conclusions

In conclusion, the current version of the Gin system provides a large step towards allowing an increased sense of subject agency while maintaining the experiment control. It does this by allowing subjects to control their own navigation through a VE while ensuring that the subject is exposed to all experimental stimuli. In addition, the Gin system maintains the tractability of VE scenario design by requiring only minimal effort on the part of the experimenter, who provides the Gin system with a topographical abstraction of the VE, and a list of stimuli in the order that they will be seen. However, there are still many improvements to the Gin system, which can be made, particularly in the realms of evaluation, interface improvements, and future scientific work.

4.1 Evaluation

In order for the system to be truly viable, more in-depth evaluation needs to occur. We will be performing two evaluations of the Gin system in the near future. For the first evaluation, we will use the included example VE, represented as a graph as described above. By generating a large number of randomly sized and connected graphs, it should be possible to clearly demonstrate how well the Gin system works, and how likely it is to generalize to environments with different graphical representations.

In addition, we will perform an evaluation by taking the self-directed movement of subjects in an experimental environment with hand-placed stimuli, (such as the experiment described in Lance et al., 2011), and testing the Gin system's ability to place stimuli against the constant, hand-placed stimuli in the same environment.

4.2 Interface Improvements

There are several near-term changes that would greatly improve the utility of the Gin system. The primary change is upgrading the Ginterface message passing system. Currently, the Ginterface is single-threaded, and blocking, meaning that any program that sends a message to the Gin system must wait for a response before it can continue. Replacing this single-threaded interface with a multithreaded, networked interface would drastically increase the speed and ease of integrating the Gin system with a VE, thereby increasing the utility of the system.

4.3 Future Work

While this version of the Gin system begins addressing the issue of subject agency in a VE, it does not currently address subject agency beyond the control of movement in the environment. Applying the Gin system to additional environmental actions will greatly improve subject agency. In addition, in the long term it is important to begin storing additional contextual information potentially related to the user's response to a stimulus in the Gin graph. By storing this contextual information, it may be possible to use it to decrease some of the large within-subject variance that occurs in neural activity across the same stimulus. We have suggested the inclusion of a contextual graph in the Gin system, and the current Gin system is designed to allow for this expansion, increasing both the tractability of scenario development and the tractability of experimental analysis.

List of Symbols, Abbreviations, and Acronyms

ARL	U.S. Army Research Laboratory
DFS	depth-first search
FOB	Forward Operating Base
Gin	Graph-theoretic Interactive Narrative
LSA	Local Situational Awareness
OPFOR	Opposing Forces
TARDEC	U.S. Tank Automotive Research Development and Engineering Center
TOC	Tactical Operating Commander
VC	Vehicle Commander
VE	virtual environment

NO. OF COPIES	ORGANIZATION
1 (PDF only)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA 8725 JOHN J KINGMAN RD STE 0944 FORT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RESEARCH LAB IMNE ALC HRR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LT 2800 POWDER MILL RD ADELPHI MD 20783-1197

NO. OF COPIES ORGANIZATION

1 ARMY RSCH LABORATORY – HRED
RDRL HRM C A DAVISON
320 MANSCEN LOOP STE 115
FORT LEONARD WOOD MO 65473

2 ARMY RSCH LABORATORY – HRED
RDRL HRM DI
T DAVIS
J HANSBERGER
BLDG 5400 RM C242
REDSTONE ARSENAL AL 35898-7290

1 ARMY RSCH LABORATORY – HRED
RDRL HRS EA DR V J RICE
2377 GREELEY RD STE R
FORT SAM HOUSTON TX 78234-7731

1 ARMY RSCH LABORATORY – HRED
RDRL HRM DG K GUNN
BLDG 333
PICATINNY ARSENAL NJ 07806-5000

1 ARMY RSCH LABORATORY – HRED
ARMC FIELD ELEMENT
RDRL HRM CH C BURNS
THIRD AVE BLDG 1467B RM 336
FORT KNOX KY 40121

1 ARMY RSCH LABORATORY – HRED
AWC FIELD ELEMENT
RDRL HRM DJ D DURBIN
BLDG 4506 (DCD) RM 107
FORT RUCKER AL 36362-5000

1 ARMY RSCH LABORATORY – HRED
RDRL HRM CK J REINHART
10125 KINGMAN RD BLDG 317
FORT BELVOIR VA 22060-5828

1 ARMY RSCH LABORATORY – HRED
RDRL HRM AY M BARNES
2520 HEALY AVE
STE 1172 BLDG 51005
FORT HUACHUCA AZ 85613-7069

1 ARMY RSCH LABORATORY – HRED
RDRL HR MP D UNGVARSKY
POPE HALL BLDG 4709
BCBL 806 HARRISON DR
FORT LEAVENWORTH KS 66027-2302

NO. OF COPIES ORGANIZATION

1 ARMY RSCH LABORATORY – HRED
RDRL HRM DQ M R FLETCHER
AMSRD NSC WS E BLDG 3 RM 341
NATICK MA 01760-5020

1 ARMY RSCH LABORATORY – HRED
RDRL HRM AT J CHEN
12350 RESEARCH PKWY
ORLANDO FL 32826-3276

1 ARMY RSCH LABORATORY – HRED
RDRL HRM AT C KORTENHAUS
12350 RESEARCH PKWY
ORLANDO FL 32826

1 ARMY RSCH LABORATORY – HRED
RDRL HRM AS C MANASCO
SIGNAL TOWERS
BLDG 29808A RM 303
FORT GORDON GA 30905-5233

1 ARMY RSCH LABORATORY – HRED
RDRL HRM CU
6501 E 11 MILE RD MS 284
BLDG 200A 1ST FL RM A1117
WARREN MI 48397-5000

1 ARL FIRES CTR OF EXCELLENCE
FIELD ELEMENT
RDRL HRM AF C HERNANDEZ
3040 NW AUSTIN RD RM 221
FORT SILL OK 73503-9043

1 ARMY RSCH LABORATORY – HRED
RDRL HRM AV S MIDDLEBROOKS
91012 STATION AVE
FORT HOOD TX 76544-5073

1 ARMY RSCH LABORATORY – HRED
RDRL HRM DW E REDDEN
6450 WAY ST
BLDG 2839 RM 310
FORT BENNING GA 31905-5400

NO. OF
COPIES ORGANIZATION

1 (CD only) ARMY G1
DAPE MR B KNAPP
300 ARMY PENTAGON RM 2C489
WASHINGTON DC 20310-0300

1 U.S. ARMY TACOM
ATTN AMSRD TAR R MS 157 V PAUL
6501 E ELEVEN MILE RD.
WARREN, MI 48397-5000

ABERDEEN PROVING GROUND

6 DIR USARL
RDRL HR
L ALLENDER
T LETOWSKI
RDRL HRM
P SAVAGE-KNEPSHIELD
RDRL HRS D
B AMREIN
RDRL HRS C
B LANCE (1 PDF, 1 HC)
K OIE